A Benchmarking Framework for Data-Driven Compressive Sensing

Márk Lakatos-Tóth, Computer Science
Mentor: Fengbo Ren, Assistant Professor
School of Computing, Informatics, and Decision Systems Engineering

Introduction

Compressive sensing is an information acquisition technique that asserts that it is possible to recover certain signals from fewer samples than traditional methods use. This is possible because many signals are sparse in some basis and therefore can have a concise representation. Compressive Sensing is all about solving an inverse problem:

x: original signal y: measurement

$$y = Ax$$

$$\text{Where } \dim(y) = m \ll n = \dim(x)$$

$$x = ?$$

The problem is underdetermined, so we need to give constraints of x to solve it.

How should we choose A? What reconstruction method should we choose?

Current compressive sensing algorithms can be divided into two categories: model-based methods that usually use expert knowledge and rely on that the signal is sparse in some basis, and data-driven methods that use training data to try to learn from the structure within the data.

Objective

The objective of the research is to develop a benchmarking framework with a unified API and benchmarks to allow researchers in this field to quantitatively evaluate new algorithms. To achieve fair and extensive comparisons, all data-driven methods are implemented in PyTorch and TensorFlow, and are compared to state-of-the-art model-based methods that are implemented in MATLAB, with a variety of dataset and parameter setups. The CSGM and CSGAN algorithms were reimplemented as part of this framework, based on the provided TensorFlow implementations. The fair evaluation of these new algorithms could ultimately lead to improvements in the practical applications of Compressive Sensing.

Examples of practical applications of Compressive Sensing

- Pixel imaging
- Accelerating Magnetic Resonance Imaging (MRI)
- Wireless tele-monitoring
- Cognitive radio communications

Algorithms included in the framework

- **Data-driven algorithms**
  - LSTA-Net
  - LDAMP
  - ReconNet
  - LAPRAN
  - CSGM
  - CSGAN
- **Model based algorithms**
  - L1 (Lasso)
  - TVAL-3
  - DAMP
  - NLR-CS

Implementation Status

The objective of this research was to reimplement the CSGM and CSGAN algorithms as part of this benchmarking framework, based on the provided TensorFlow implementations. All the algorithms of the framework have been successfully reimplemented and integrated into the framework. It was ensured that they can also reproduce the results reported in their respective original research papers. The dataset and benchmark parameter setups have been created, and the benchmarks and evaluations are currently running with these setups.

CSGAN

The following algorithm describes the implementation of CSGAN using a sensing matrix as measurement function

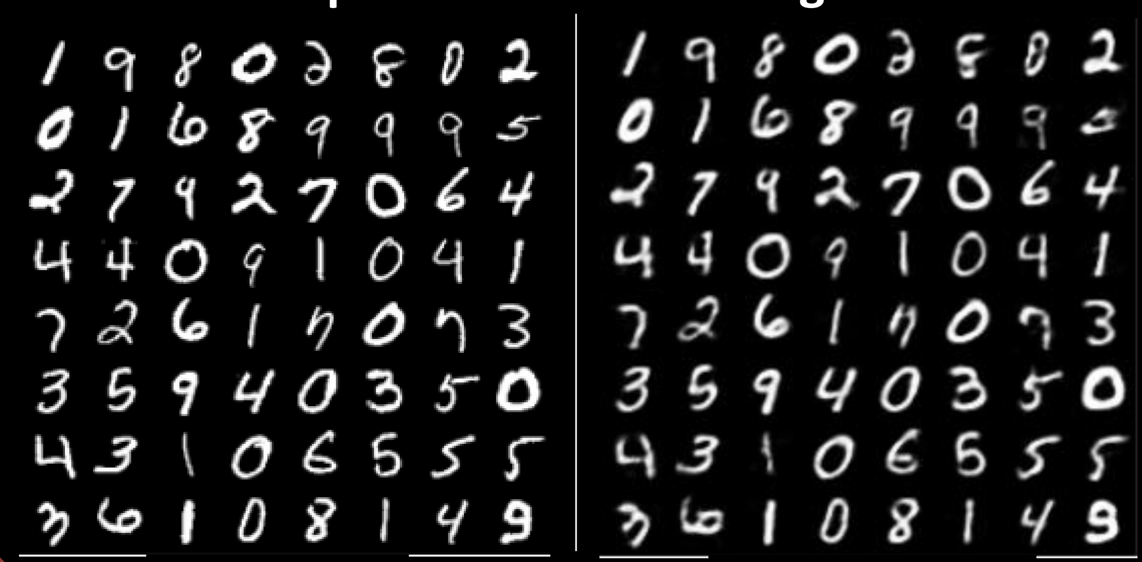
Algorithm 1 Compressed Sensing with Meta Learning

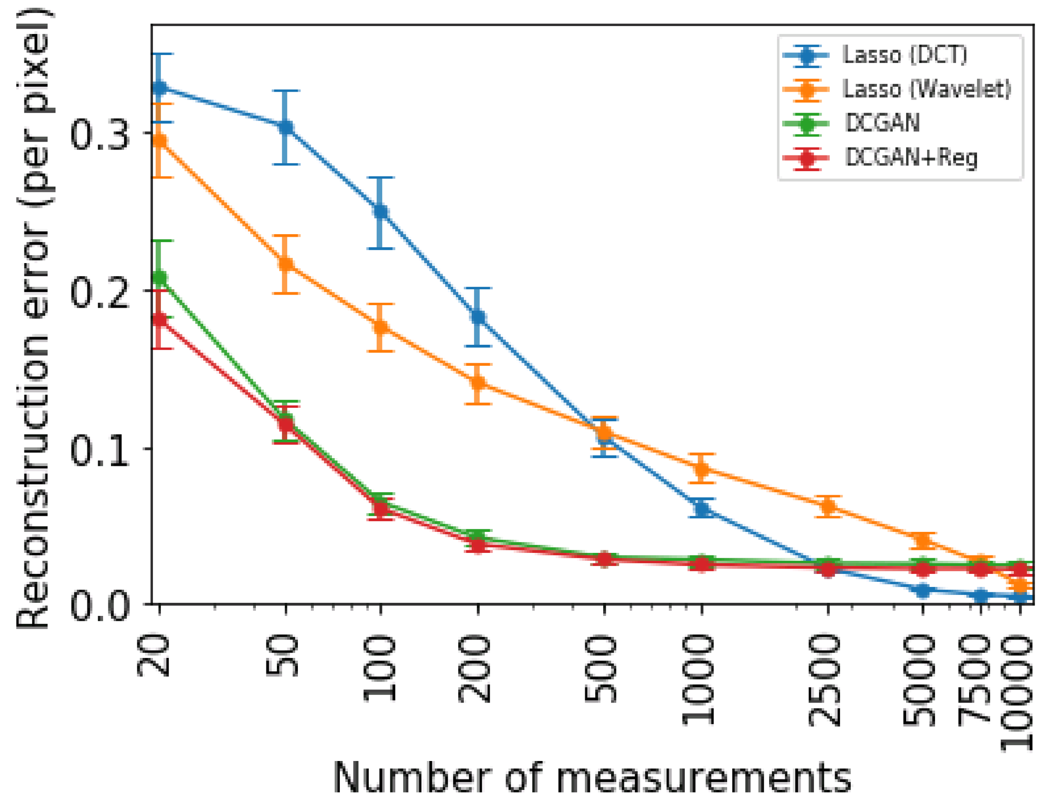
Input: minibatchs of data $\{\boldsymbol{x}_i\}_{i=1}^N$, random matrix $\mathbf{F}$, generator G_θ, learning rate α, number of latent optimisation steps T

repeat
- Initialize generator parameters θ
- **for** $i = 1$ **to** N **do**
  - Measure the signal $\boldsymbol{m}_i \leftarrow \mathbf{F}\,\boldsymbol{x}_i$
  - Sample $\hat{\boldsymbol{z}}_i \sim p_{\boldsymbol{z}}(\boldsymbol{z})$
  - **for** $t = 1$ **to** T **do**
    - Optimise $\hat{\boldsymbol{z}}_i \leftarrow \hat{\boldsymbol{z}}_i - \frac{\partial}{\partial \boldsymbol{z}} E_\theta(\boldsymbol{m}_i, \hat{\boldsymbol{z}}_i)$
  - **end for**
- **end for**
- $\mathcal{L}_G = \frac{1}{N}\sum_{i=1}^{N} E_\theta(\boldsymbol{m}_i, \hat{\boldsymbol{z}}_i)$
- Compute $\mathcal{L}_F$ using eq. 12
- Update $\theta \leftarrow \theta - \frac{\partial}{\partial \theta}(\mathcal{L}_G + \mathcal{L}_F)$

until reaches the maximum training steps

An example reconstruction using the reimplemented CSGAN algorithm





Results for CSGM on CelebA

Future Work

- New hyperparameter and dataset setups can be designed to create even more extensive benchmarks and comparisons
- In the future, new methods could be added to the framework

Acknowledgements

I would like to thank my mentor, Dr. Fengbo Ren, as well as Zhikang Zhang for their support in this project. I would also like to thank the Fulton Undergraduate Research Initiative for funding this project.

FURI

Ira A. Fulton Schools of Engineering
Arizona State University